

FILE ID**COBCVTDD

c 11

The diagram consists of several columns of symbols. On the far left, there are two columns of 'L' symbols. To their right is a single column of 'I' symbols. Further to the right is a series of 'S' symbols arranged in a descending staircase pattern, where each row contains one less 'S' than the row above it. The entire pattern is contained within a rectangular frame.

(2)	49	HISTORY	: Detailed Current Edit History
(3)	62	DECLARATIONS	
(4)	110	COB\$CVTDQ_R8	

```
0000 1 .TITLE COB$CVTDQ_R8 COBOL Convert Double Floating to Quadword
0000 2 .IDENT /1-006/ ; File: COBCVTDQ.MAR
0000 3
0000 4
0000 5 ****
0000 6 *
0000 7 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9 * ALL RIGHTS RESERVED.
0000 10 *
0000 11 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16 * TRANSFERRED.
0000 17 *
0000 18 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20 * CORPORATION.
0000 21 *
0000 22 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24 *
0000 25 *
0000 26 ****
0000 27 *
0000 28 .FACILITY: COBOL SUPPORT
0000 29 ++
0000 30 .ABSTRACT:
0000 31     This module contains the routine that converts double floating
0000 32     numbers to quadword.
0000 33
0000 34
0000 35 --
0000 36
0000 37 .VERSION: 1
0000 38
0000 39 .HISTORY:
0000 40
0000 41 .AUTHOR:
0000 42     Marty Jack, 14-Mar-1979
0000 43
0000 44 .MODIFIED BY:
0000 45
0000 46
0000 47 ;
```

0000 49 .SBTTL HISTORY ; Detailed Current Edit History
0000 50
0000 51
0000 52 : Edit History for Version 1 of COBCVTDO
0000 53 :
0000 54 : 1-001 - Original. MLJ 14-Mar-1979
0000 55 : 1-002 - Correct problem with high order longword. MLJ 22-Mar-1979
0000 56 : 1-003 - Correct round towards zero problem. PDG 12-Jul-1979
0000 57 : 1-004 - Make external references explicit. RKR 17-JULY-79
0000 58 : 1-005 - Change all references to FOR\$CNV_OUT_E to COB\$CNVOUT.
0000 59 : RKR 27-SEPT-79
0000 60 : 1-006 - Cosmetic changes. 18-OCT-79

```
0000 62      .SBTTL DECLARATIONS
0000 63
0000 64 : INCLUDE FILES:
0000 65 : $DSCDEF
0000 66 :
0000 67
0000 68
0000 69 :
0000 70 : EXTERNAL SYMBOLS:
0000 71     .DSABL GBL          : Prevent undefined symbols from
0000 72             : being automatically global
0000 73
0000 74     .EXTRN COB$CNVOUT   : E-format conversion
0000 75     .EXTRN LIB$STOP      : Error halt
0000 76     .EXTRN OTSS_FATINTERR : Fatal internal error code
0000 77
0000 78 :
0000 79
0000 80 :
0000 81 : MACROS:
0000 82     NONE
0000 83 :
0000 84
0000 85 :
0000 86 : PSECT DECLARATIONS:
0000 87     .PSECT _COB$CODE    PIC, SHR, LONG, EXE, NOWRT
0000 88
0000 89 :
0000 90 : EQUATED SYMBOLS:
0000 91     NONE
0000 92 :
0000 93
0000 94 :
0000 95 : OWN STORAGE:
0000 96 :
0000 97 :+
0000 98 : The following constant has the value 2**32. It is used for scaling
0000 99 : the high 32 bits and for compensating for unsigned arithmetic.
0000 100 :-
0000 101 BIAS: .PACKED 4294967296
0000 102 :+
0000 103 : The following constant is 2**32-1. It's subtracted from negative numbers,
0000 104 : to get a DIVP to give the correct high-order longword.
0000 105 :-
0000 106 BIAS_1: .PACKED 4294967295
0000 107 BIAS_DIGITS=10
0000 108 :
```

6C 29 67 49 29 04

5C 29 67 49 29 04
0000000A

000C 110 .SBTTL COB\$CVTDQ_R8
000C 111
000C 112 :++
000C 113 : FUNCTIONAL DESCRIPTION:
000C 114 : Converts a double floating number to quadword.
000C 115
000C 116
000C 117 : CALLING SEQUENCE:
000C 118
000C 119 JSB COB\$CVTDQ_R8 (scale.rl.v, src.rd.r, dst.wq.r)
000C 120
000C 121 Arguments are passed in R6, R7, and R8.
000C 122
000C 123 : INPUT PARAMETERS:
000C 124
000C 125 : SCALE.rl.v The power of ten by which the internal
000C 126 : representation of the source must be
000C 127 : multiplied to scale the same as the
000C 128 : internal representation of the dest.
000C 129 : SRC.rd.r The number to be converted
000C 130
000C 131 : IMPLICIT INPUTS:
000C 132
000C 133 All of the trap bits in the PSL are assumed off.
000C 134
000C 135 : OUTPUT PARAMETERS:
000C 136
000C 137 : DST.wq.r The place to store the converted number
000C 138
000C 139 : IMPLICIT OUTPUTS:
000C 140
000C 141 : NONE
000C 142
000C 143 : FUNCTION VALUE:
000C 144
000C 145 1 = SUCCESS, 0 = FAILURE
000C 146
000C 147 : SIDE EFFECTS:
000C 148
000C 149 Destroys registers R0 through R8.
000C 150
000C 151 :--
000C 152
000C 153
000C 154 COB\$CVTDQ_R8::
5E 30 C2 000C 155 SUBL2 #48,SP : Allocate temp space
6E 67 70 000F 156 MOVD (R7), (SP) : Get input number
0012 157
0012 158 : Make a descriptor for the temporary string.
0012 159 :
08 AE 9F 0012 160 PUSHAB 8(SP) : Address = space reserved
7E 01 90 0015 161 MOVB #DSC\$K_CLASS_S,-(SP) : Class = static
7E 0E 90 0018 162 MOVB #DSC\$K_DTYPE_T,-(SP) : Data type = ASCII string
7E 26 B0 001B 163 MOVW #38,-(SP) : Length in bytes
001E 164 :
001E 165 : Call COB\$CNVOUT.
001E 166 :

					PUSHL #31	: Digits in fraction
					PUSHAB 4(SP)	: Output string descriptor
					PUSHAB 16(SP)	: Number to convert
					CALLS #3,G^COB\$CNVOUT	: Call conversion routine
					BLBC R0,20\$: Should never fail
					0030 173 :	
					0030 174 :	Convert the exponent and correct for scale factor.
					0030 175 CVTSP #2,51(SP),#2,(SP)	: Make packed exponent
					0036 176 CVTPL #2,(SP) R6	: Make longword exponent
					003A 177 MOVAB -31(R6)[R0],R6	: Correct for fraction size and scale
					003F 178 :	
					003F 179 :	Convert the fraction to packed.
					003F 180 :	
					003F 181 MOVB 16(SP),18(SP)	: Move sign over "..."
					0044 182 CVTSP #31,18(SP),#31,(SP)	: Make packed fraction
					004A 183 ASHP R6,#31,(SP),#0,#19,16(SP)	: Scale to temporary (also clears R0)
					0052 184 BVS 10\$: Branch if won't fit in 19 digits
					0052 185 CVTPL #19,(R3),(R8)	: Convert to longword (also clears R0)
					0058 186 BVS 11\$: Can't fit in 32 bits
					0058 187 ASHL #31,(R8)+,(R8)	: Spread sign bit
					005F 188 INCL R0	: Indicate success, R0 = 1
					0061 189 ADDL2 #56,SP	: Delete stack temps
					0064 190 RSB	
					0065 191 10\$: :	
					0065 192 :	
					0065 193 :	
					0065 194 :	Come here if the packed number won't fit in 32 bits.
					0065 195 :	Divide by 2**32 to get the high 32 bits of the quadword.
					0065 196 :	
					0065 197 11\$: :	BLBC 9(R1),13\$: Skip if positive
					0069 198 SUBP4 #BIAS_DIGITS,BIAS 1,#19,(R1) : Make more negative	
					006F 199 13\$: :	DIVP #BIAS_DIGITS,BIAS,#19,16(SP),#19,(SP) : Divide by 2**32
					0077 200 CVTPL #19,(SP),4(R8)	: Convert and store high bits (also clears R0)
					007D 201 BVS 12\$: Number too large for a 64-bit integer
					007D 202 INCL R0	: Indicate success, R0 = 1
					007F 203 ADDL2 #56,SP	: Delete stack temps
					0084 204 RSB	: Return
					0085 205 :	
					0085 206 :	
					0085 207 :	Come here on failure of COB\$CNVOUT. This should never happen.
					0085 208 :	
					0085 209 20\$: :	PUSHL #OTSS_FATINTERR : OTS fatal error message
					008B 210 CALLS #1,G^IB\$STOP : Signal and don't return	
					0092 211 :	
					0092 212 .END	

BIAS
 BIAS_1
 BIAS_DIGITS
 COB\$ENVOUT
 COB\$CVTDQ R8
 DSC\$K_CLASS_S
 DSC\$K_DTYPE_T
 LIB\$STOP
 OTSS_FATINTERR

	00000000	R	02
=	00000006	R	02
	0000000A		
	*****	X	00
	0000000C	RG	02
=	00000001		
=	0000000E		
	*****	X	00
	*****	X	00

+-----+
 ! Psect synopsis !
 +-----+

PSECT name

PSECT name	Allocation	PSECT No.	Attributes	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE
. ABS .	00000000	(0.)	00 (0.)	NOPIIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT
\$ABSS	00000000	(0.)	01 (1.)	NOPIIC	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT
_COB\$CODE	00000092	(146.)	02 (2.)	PIC	USR	CON	REL	LCL	SHR	EXE	RD	NOVEC LONG

+-----+
 ! Performance indicators !
 +-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	32	00:00:00.04	00:00:00.91
Command processing	135	00:00:00.36	00:00:03.75
Pass 1	140	00:00:01.28	00:00:07.08
Symbol table sort	0	00:00:00.11	00:00:00.77
Pass 2	50	00:00:00.39	00:00:02.40
Symbol table output	2	00:00:00.01	00:00:00.01
Psect synopsis output	3	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	364	00:00:02.21	00:00:14.94

The working set limit was 1050 pages.

9010 bytes (18 pages) of virtual memory were used to buffer the intermediate code.

There were 10 pages of symbol table space allocated to hold 138 non-local and 5 local symbols.

212 source lines were read in Pass 1, producing 10 object records in Pass 2.

8 pages of virtual memory were used to define 7 macros.

+-----+
 ! Macro library statistics !
 +-----+

Macro library name

_S255\$DUA2B:[SYSLIB]STARLET.MLB;2

Macros defined

4

190 GETS were required to define 4 macros.

There were no errors, warnings or information messages.

MACRO/ENABLE=SUPPRESS ON/DISABLE=(GLOBAL,TRACEBACK)/LIS=LIS\$:COBCVTDO/OBJ=OBJ\$:COBCVTDO MSRC\$:COBCVTDO/UPDATE=(ENH\$:COBCVTDO)

0061 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

COBCUTRPO
LIS

COBCUTQP
LIS

COBCUTQD
LIS

COBCANCEL
LIS

COBACCTIM
LIS

COBCUTPQ
LIS

COBCUTRFQ
LIS

COBCUTQP
LIS

COBCUTROP
LIS

COBCALL
LIS

COBCUTFQ
LIS

COBCUTRDQ
LIS

COBCUTQD
LIS

COBACCEPT
LIS